

Oracle 数据库 10g

网络数据模型

Oracle 技术白皮书

2003年12月

目录

1 引言	1
2 设计目标和体系结构.....	2
3 使用网络数据模型的主要步骤.....	3
3.1 网络建模.....	3
3.2 网络分析.....	4
4 网络建模.....	4
5 元数据和模式.....	5
6 网络 Java 表示和 API.....	6
7 利用 SQL 和 PL/SQL 进行网络创建.....	7
7.1 创建逻辑网络.....	8
7.2 创建空间网络.....	8
7.2.1 创建 SDO 空间网络.....	9
7.2.2 创建线性参照空间网络.....	9
7.2.3 创建拓扑空间网络.....	10
8 利用 Java API 进行网络创建.....	10
9 利用 Java API 进行网络分析.....	12
10 网络数据的参考完整性.....	13
11 网络约束.....	14
12 路径表示.....	15
13 要求	17
14 结论	17

1 引言

许多应用要求能够对相关的对象之间的关系进行建模和分析。网络（或图形）就是这样的一种建模表示。在一个网络表示中存在三种元素：节点、连接和路径。相关的对象称为节点，节点之间的关系称为连接。路径是有序的连接列表，其中没有重复的节点或连接。注意对网络建模只需要连通性信息。此外，在连接和节点中引进了成本（或权重）的概念（如果在分析期间需要考虑的话）。

图 1 显示了纽约市的道路网络。

图 1：纽约市道路网络（60384 个节点、151962 条连接。来源：来自 NavTech 的 NavStreets）



Oracle 数据库 10g 中的 Oracle Spatial 提供一个网络数据模型使您能对网络进行建模和分析。该网络数据模型包含两个组件：网络数据库模式和 Java API。网络模式包含网络元数据、节点表、连接表和路径元数据表以及和一个 PL/SQL 程序包 (SDO_NET)。Java API 实现了网络表示和网络分析。

该网络数据模型可以表示逻辑网络和空间网络。逻辑网络没有与它们相关的任何空间信息。空间网络包含能够利用任何 Oracle Spatial 几何数据 (SDO 几何数据、线性参照几何数据或拓扑几何数据) 表示的空间信息。此外，可以通过将节点组织在不同的层次水平上来表示层次关系。

Oracle Spatial 网络数据模型提供了一个开放和持续的网络数据模型，简化了网络数据管理和分析，分离了连通性和应用程序信息，以便应用程序可以关注特定的应用知识，并与 Oracle Spatial 技术集成来进行空间信息管理。

本文其余的部分将讨论网络数据模型的方法、网络元数据和模式对象、网络 PL/SQL 和 Java API。它还包含了使用信息和网络创建、加载和分析的代码摘录。

2 设计目标和体系结构

网络数据模型有以下设计目标：

- 为网络应用程序提供一个开放和通用的网络数据模型。

网络数据模型信息存储在数据库中的表中。可以执行标准的 SQL 查询、PL/SQL 函数和过程。网络数据模型只采集通用的连通性信息，从而清楚地将连通性和应用程序信息分离。网络约束机制使得与应用程序相关的约束能够指导网络分析，而无需了解应用程序环境。

- 实现空间信息支持。

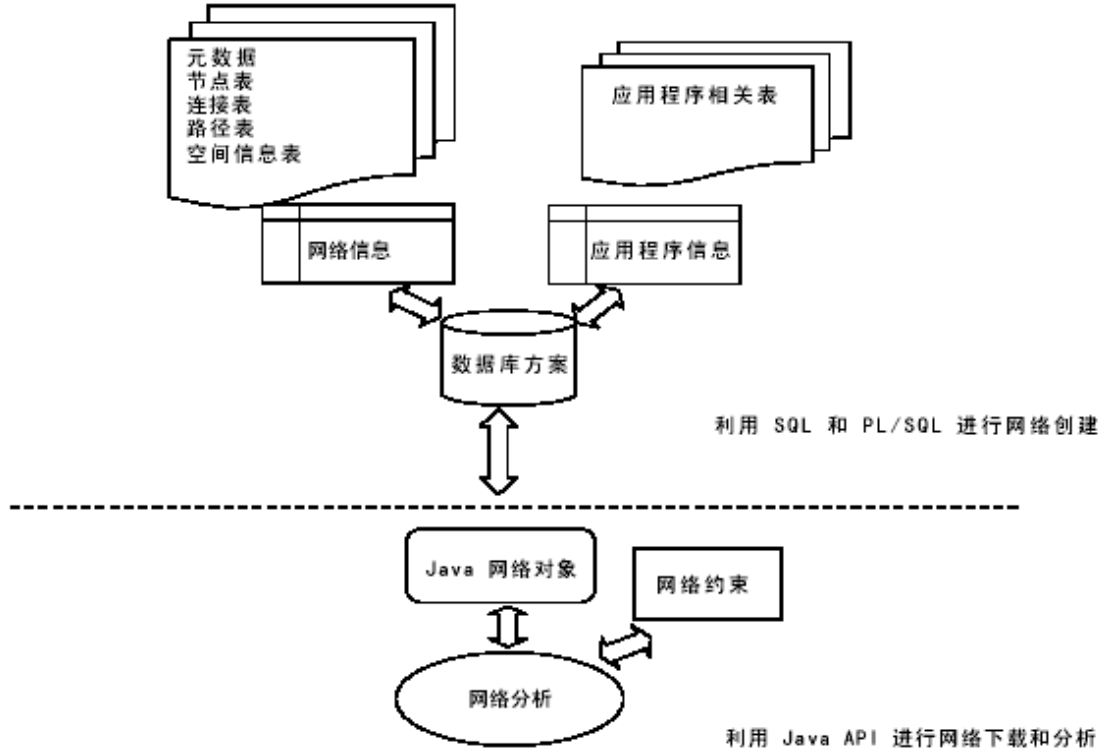
空间信息可以与网络关联。几何数据可按任何一种 Oracle Spatial 格式存储：SDO几何数据、线性参照几何数据、或拓扑几何数据。

- 提供了一个 2 层或 n 层的体系结构。

网络信息在关系数据库中存储和管理（通过 PL/SQL 和 SQL），网络表示、网络加载和网络分析在客户端或应用程序层通过 Java API 进行。

图 2 显示了网络数据模型的体系结构。

图 2：网络数据模型体系结构



3 使用网络数据模型的主要步骤

使用网络数据模型时有两个主要的步骤：网络建模和网络分析。

3.1 网络建模

网络建模将网络应用程序转换成网络表示（节点、连接和路径），以便能够执行网络分析。应用程序提取连通性信息并保留网络元素和应用程序特性之间的映射关系。

例如，一个道路网络包含高速公路和城市，它们被分别映射为连接和节点。距离或旅行时间可以是与每一条连接相关的成本。当进行路径选择分析时，最短的（或最快的）路径返回在一个有序的连接列表中。连接列表被映射回应用程序域，以获得详细的方向提示。

用户可以采用以下方法之一进行网络建模：

- 利用 SQL 或 PL/SQL 在数据库中创建网络。

网络数据模型提供了一个 PL/SQL 程序包 (SDO_NET)，以在数据库中创建持续的网络模型。还存储了网络元数据，因为它提供了关于网络的信息。

- 利用 Java API 在客户端层创建网络。

还可以利用网络数据模型提供的 Java API 来创建网络。该 API 包含网络元素的 Java 表示(接口和类)，并且它可以用来创建 Java 网络表示。Java 网络对象是临时的，可以存储在数据库中。

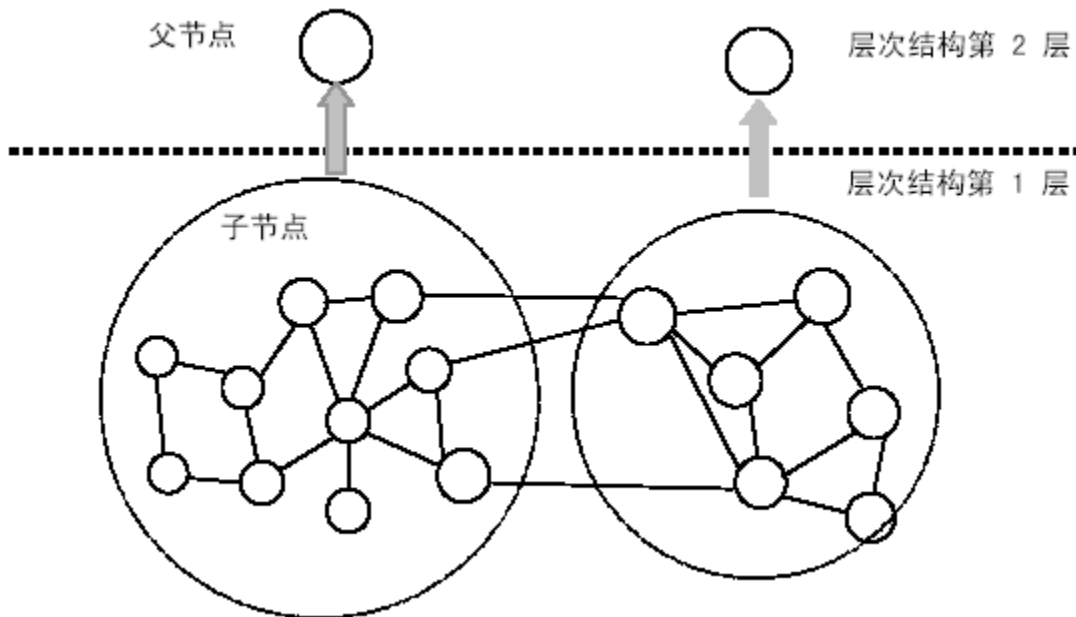
3.2 网络分析

网络数据模型中的网络分析是在客户端或应用程序层通过 Java API 执行的。当从数据库加载网络时，将创建一个 Java 网络对象并为分析做好准备；不过，在分析它之前，您应当验证网络（元数据和参考完整性）。

4 网络建模

网络数据模型能够用来对逻辑网络（无空间信息）和空间网络（有空间信息）建模。空间网络支持以下 Oracle Spatial 类型：SDO 几何数据、LRS 几何数据和拓扑几何数据。

图 3：网络中的层次关系



空间和逻辑网络都可以通过层次关系（父子）进行建模，如图 3 所示。

5 元数据和模式

网络元数据包含了关于网络的信息。必须在USER_SDO_NETWORK_METADATA 视图中为所创建的每个网络插入信息。该视图包含了以下信息的列 —一个列可以包含空值如果它不能适用到某个特定网络：

- NETWORK （网络名称）
- NETWORK_CATEGORY （'LOGICAL' 或 'SPATIAL'）
- GEOMETRY_TYPE （对空间网络，可设成 'SDO_GEOMETRY'、'LRS_GEOMETRY' 或 'TOPO_GEOMETRY'；对逻辑网络，设成 NULL）
- NODE_TABLE_NAME
- NODE_GEOM_COLUMN
- LINK_TABLE_NAME
- LINK_GEOM_COLUMN
- LRS_TABLE_NAME
- LRS_GEOM_COLUMN
- PATH_TABLE_NAME
- PATH_GEOM_COLUMN
- PATH_LINK_TABLE_NAME
- LINK_COST_COLUMN
- NODE_COST_COLUMN
- LINK_DIRECTION （'DIRECTED' 或 'UNDIRECTED'）
- NO_OF_HIERARCHY_LEVELS

此外，必须为每个网络创建和填充下列表格，以包含关于节点、连接和路径的信息：

- 节点表

- 连接表
- 路径表（只有在网络包含路径时）
- 路径和连接表（只有在创建了路径表时；为网络中的每条路径中的每条连接包含一行）

6 网络 Java 表示和 API

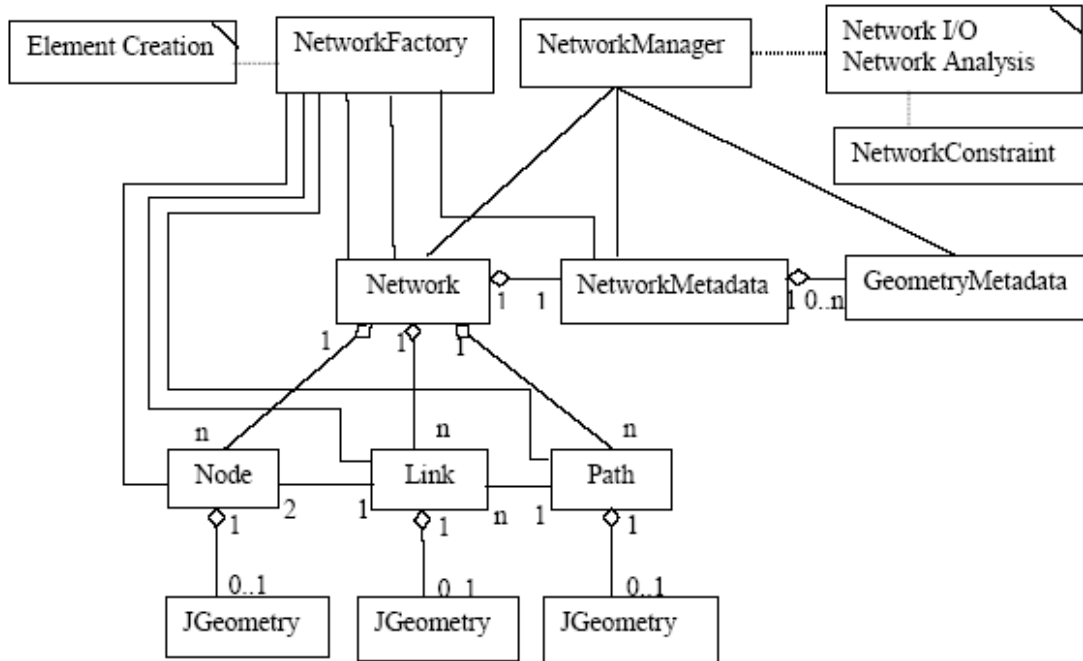
网络数据模型提供了一个用于网络表示和分析的 Java API。网络数据模型的 Java 客户端接口包含以下类和接口：

- NetworkMetadata：网络元数据
- GeometryMetadata：空间网络的几何元数据
- Network：网络表示
- Node：网络节点
- Link：网络连接
- Path：网络路径
- NetworkDataException：网络数据模型异常
- Jgeometry：Java SDO几何类
- NetworkConstraint：网络分析的搜索约束条件

图 4 显示了主要的类和接口之间的关系。

图 4：网络数据模型的 Java 类和接口

*程序包：oracle.spatial.network 和
oracle.spatial.geometry*



用以下两个 Java 类进行网络管理和网络分析：

- NetworkFactory：网络创建、网络元素创建。
- NetworkManager：网络加载、存储和分析

7 利用 SQL 和 PL/SQL 进行网络创建

网络数据模型提供了一些 PL/SQL 过程（程序包 SDO_NET），以简化网络创建和管理。可以利用预先定义的表名称、列名称和默认设置来创建网络表。当创建网络时，网络元数据视图会自动更新。网络表名称、列名称和元数据的默认值如下：

- 节点表名称：<network>_NODES\$
- 连接表名称：<network>_LINKS\$
- 路径表名称：<network>_PATHS\$
- 路径连接表名称：<network>_PLINKS\$
- 节点几何数据列：GEOMETRY
- 连接几何数据列：GEOMETRY

- 路径几何数据列：GEOMETRY
- 连接成本列：COST
- 节点成本列：null（无成本的节点）或 COST（有成本的节点）

本部分描述了创建逻辑网络和空间网络的主要步骤。不过，关于详细的指导和说明，请参见 *Oracle 空间拓扑和网络数据模型手册*。

7.1 创建逻辑网络

在逻辑网络的网络元数据中，NETWORK_CATEGORY 为 LOGICAL，GEOMETRY_TYPE 为 null。

请按照以下主要步骤创建逻辑网络：

1. 创建网络表并插入网络元数据。例如：

```
EXEC SDO_NET.CREATE_LOGICAL_NETWORK(
  'LOG_NET', -- 网络名称
  1, -- 网络层次
  true, -- 有向连接
  false, -- 没有成本
);
```

2. 填充节点表和连接表。

3. 验证网络。例如

```
SDO_NET.VALIDATE_NETWORK('LOG_NET');
```

7.2 创建空间网络

空间网络包含以下三种类型之一的几何数据：

- SDO几何数据：非线性参照几何数据（即不包含度量维）的 SDO_GEOMETRY 类型的对象
- LRS 几何数据：线性参照几何数据（即包含度量维）的 SDO_GEOMETRY 类型的对象
- 拓扑几何数据：SDO_TOPO_GEOMETRY 类型的对象

7.2.1 创建 SDO 空间网络

在具有 SDO 几何数据的空间网络的网络元数据中，NETWORK_CATEGORY 为 SPATIAL，GEOMETRY_TYPE 为 SDO_GEOMETRY。

请按照以下主要步骤创建具有 SDO 几何数据的空间网络：

1. 创建网络表并插入网络元数据。例如：

```
EXEC SDO_NET.CREATE_SDO_NETWORK(  
  'SDO_NET', --网络名称  
  
  1, --网络层次  
  
  true, --有向连接  
  
  false, --没有成本  
  
);
```

2. 填充节点表和连接表。

3. 为节点表和连接表、路径表以及路径和连接表（如果将使用路径）插入几何元数据。

4. 验证网络。例如：

```
SDO_NET.VALIDATE_NETWORK('SDO_NET');
```

7.2.2 创建线性参照空间网络

在具有线性参照几何数据的空间网络的网络元数据中，NETWORK_CATEGORY 为 SPATIAL，GEOMETRY_TYPE 为 LRS_GEOMETRY。

请按照以下主要步骤创建具有线性参照几何数据的空间网络：

1. 创建网络表并插入网络元数据。例如：

```
EXEC SDO_NET.CREATE_LRS_NETWORK(  
  'LRS_NET', --网络名称  
  
  'LRS_GEOM_TABLE', -- 线性参照几何数据的表名  
  
  'LRS_GEOM_COLUMN', -- 线性参照几何数据的列名  
  
  1, --网络层次  
  
  true, --有向连接  
  
  false, --没有成本  
  
);
```

2. 填充节点表和连接表。
3. 为节点表和连接表、路径表以及路径和连接表（如果将使用路径）插入几何元数据。
4. 验证网络。例如：

```
SDO_NET.VALIDATE_NETWORK('LRS_NET');
```

7.2.3 创建拓扑空间网络

在具有拓扑几何数据的空间网络的网络元数据中，NETWORK_CATEGORY 为 SPATIAL，GEOMETRY_TYPE 为 TOPO_GEOMETRY。

请按照以下主要步骤创建具有拓扑几何数据的空间网络：

1. 创建网络表并插入网络元数据。例如：

```
EXEC SDO_NET.CREATE_TOPO_NETWORK(  
  'TOPO_NET', -- 网络名称  
  1, -- 网络层次  
  true, -- 有向连接  
  false, -- 没有成本  
);
```

2. 填充节点表和连接表。
3. 为节点表和连接表、路径表以及路径和连接表（如果将使用路径）插入几何元数据。
4. 验证网络。例如：

```
SDO_NET.VALIDATE_NETWORK('TOPO_NET');
```

8 利用 Java API 进行网络创建

除了使用 PL/SQL 和 SQL 之外，还可以使用网络数据模型 Java API 来创建网络。有 Java 类和接口表示在应用程序或客户端层的网络和网络元素。要创建 Java 网络，用网络元数据创建一个空的网络对象，并添加节点、连接和可选路径。一旦创建了 Java 网络对象，您就可以在其上执行网络分析。还可以在数据库中存储 Java 网络对象。

以下代码摘录创建了一个网络，执行了一些分析，并在数据库中存储了该网络：

```

...
// create an empty logical network
Network logicalNetwork
= NetworkFactory.createLogicalNetwork(
"LOGICAL_NET",
1, // no of hierarchy levels
true // directed link?
);
// create nodes and add them to the network
logicalNetwork.addNode(NetworkFactory.createLogicalNode(
1, // node ID
"N1" // Node Name
);
logicalNetwork.addNode(NetworkFactory.createLogicalNode(
2, // node ID
"N2" // Node Name
);
...
// create links and add them to the network
logicalNetwork.addLink(NetworkFactory.createLogicalLink(
1, // link ID
"L1", // link name
logicalNetwork.getNode(1),
logicalNetwork.getNode(2),
...
// find the shortest path
Path path = logicalNetwork.shortestPath(logicalNetwork,1,2);
path.setID(1); // set Path ID
//add the path to network
logicalNetwork.addPath(path);
// save the network to database
NetworkManager.writeNetwork(con,logicalNetwork);

```

9 利用 Java API 进行网络分析

一旦网络被存储在数据库中，Java API 能够加载网络并执行网络分析。网络数据模型目前提供了以下分析功能：

- 跟踪：
 - 找到从一个给定节点可以到达的所有节点
 - 找到可以到达一个给定节点的所有节点
- 最短路径：
 - 找到起始节点和目标节点之间的最短路径
- 最小成本生成树：
 - 找到连接所有节点的最小成本生成树
- 成本内：
 - 找到一个给定节点已定成本内的所有节点
- 最近邻居：
 - 根据成本找到一个给定节点的 N 个最近节点

以下代码摘录显示了一些网络分析：

```
try {
// load a network named "LOG_NET" from database
Network network = NetworkManager.readNetwork(con, "LOG_NET");
...
// find the shortest path between startNodeID and EndNodeID
Path path =
NetworkManager.shortestPath(network, startNodeID, endNodeID);
...
// check if startNodeID can be reached by EndNodeID
if ( NetworkManager.isReachable(network, startNodeID, endNodeID) ) {
...
}
```

```
// find 10 nearest neighbors from startNodeID
Path [] pathArray =
Networkmanager.nearestNeighbors(network, startNodeID, 10);
...
// find all nodes which are within cost 20 from startNodeID
pathArray = NetworkManager.withinCost(network, startNodeID, 20.0);
// find all links in the minimum cost spanning tree of the network
Link [] linkArray = NetworkManager.mcstLinkArray(network);
} catch (NetworkDataException exp) {
}
}
```

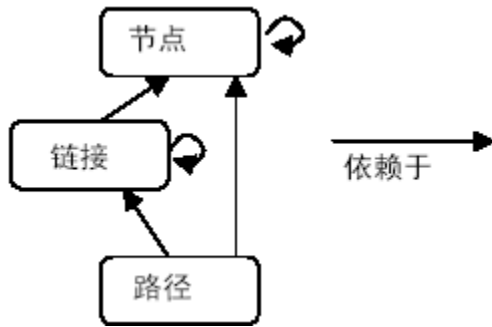
10 网络数据的参考完整性

要确保网络数据的参考完整性，可以创建参考约束，并将其应用到网络表中。可以在一个网络上创建、启用和禁用参考完整性检查。如果为网络启用了参考完整性检查，以下依赖性被强制执行：

- 节点/连接依赖性：一条连接依赖于它的起始节点和终端节点。如果删除了起始节点或终端节点，就删除了连接。如果连接所依赖的节点不存在，则不能添加连接。
- 路径/连接依赖性：路径依赖于它的连接。如果路径的任一连接被删除，则该路径也就被删除。如果路径所依赖的连接不存在，则不能添加路径。
- 层次依赖性：如果一个更高层次水平上的元素所依赖的元素不存在，则该层次的元素也不存在。

图 5 显示了一个网络的依赖性关系。您可以通过验证一个网络来检查它的网络元数据和模式，确保它和它的元素的参考完整性。

图 5：网络元素依赖性



11 网络约束

网络约束是用于网络分析的有用的过滤器。网络约束可以在非常早的阶段就快速地将搜索限制在期望的目标中。网络分析通常以一种非常特定（约束的）方式来限制可能解的数量。

网络数据模型使应用程序能够控制网络分析，而无需了解分析是如何进行的。这个特性补充了将连通性信息与应用程序信息分离的特性。网络数据模型提供了两种网络约束：系统约束和用户约束。

系统约束是对网络分析的通用约束。它们包括以下类型：

- MBR 约束：要作为有效的候选节点，节点必须在给定的 MBR 之内。
- 路径成本约束：所求路径的成本不能大于指定成本。
- 路径深度约束（连接数）：路径长度不能大于指定数量。
- 必须避免的节点和连接：不能包含在路径中的指定节点和连接。

另一方面，用户约束是依赖于应用程序的约束，这些约束可以通过 NetworkConstraint 接口实施。如果这种约束存在，那么应用程序负责提供实施并指导网络分析。

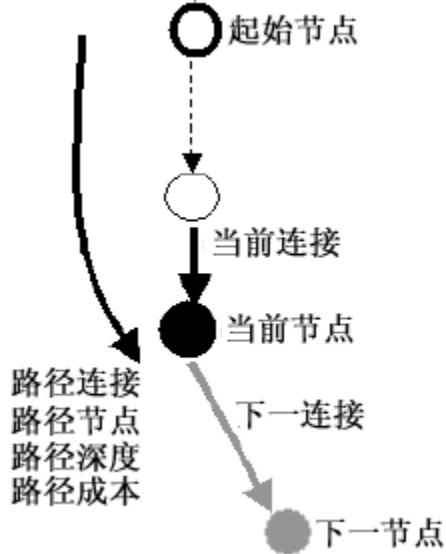
在 NetworkConstraint 中需要实施的两种方法是：

- boolean requiresPathLinks() 指示是否需要所有的路径连接和节点来实施约束。
- boolean isSatisfied(AnalysisInfo info)，检查是否满足约束条件。AnalysisInfo 被传递给应用程序，它包含关于当前搜索的以下信息：起始节点、当前节点、下一节点、当前连接、下一连接、当前深度（连接数）、当前成本。如果 requiresPathLinks() 为真，它还包含路径

连接和节点。

图 6 显示了网络约束的分析信息。

图 6：网络约束的分析信息



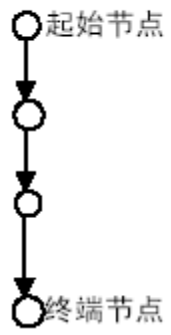
12 路径表示

一条简单路径是一个不含重复节点的有序的连接列表。它通常是一次路径计算的结果。不过，在一些应用中，一条路径是对一组拥有起始节点和终端节点的简单路径的一种抽象表示。这种类型的路径在网络数据模型中被称为复杂路径。复杂路径可被用于路径计算来识别一组特定的简单路径，如最短路径或长度小于一个给定长度阈值的路径组。在路径和连接表中不需要提供复杂路径的连接，因为网络、起始节点和终端节点为路径分析提供了足够的信息。

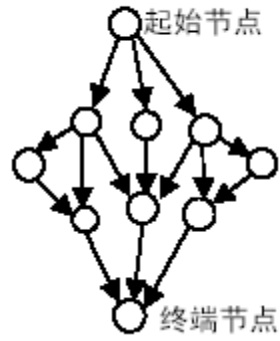
图 7 显示了简单路径和复杂路径。

图 7：简单路径和复杂路径

简单路径



复杂路径



13 要求

网络数据模型包含在 Oracle 数据库 10g 的 Oracle Spatial 中。具体来说，网络数据模型需要 SDO_NETWORK PL/SQL 程序包、网络模型 Java API（程序包：oracle.spatial.network），Oracle SDO 几何数据 Java API（程序包：oracle.spatial.geometry）。

数据库连接还需要 Oracle JDBC 驱动程序。

14 结论

网络数据模型作为 Oracle 数据库 10g 中的 Oracle Spatial 中的一个特性提供了对网络应用程序进行建模和分析的功能。网络数据模型是一种开放和一般的网络模型，在数据库中以对象关系的形式表示网络，并在客户端和应用程序层将网络表示为 Java 对象。2 层或 n 层的体系结构利用了数据库和 Java 功能。网络数据模型的主要目标是清楚地将应用程序逻辑和属性与一般的网络信息分离开，并简化网络管理和分析，以便网络应用程序能够专注于它们的应用程序知识。网络数据模型与 Oracle Spatial 紧密集成在一起，以扩展 Spatial 的空间信息管理功能，并使其成为一个强大的网络应用程序建模和分析平台。

15 参考文献

关于技术使用和参考文献信息，请参见 Oracle 数据库 10g 文档集中的以下手册：

- *Oracle Spatial 拓扑和网络数据模型*
- *Oracle Spatial 用户指南和参考*



版权所有 © 2003。Oracle 公司
保留所有权利

Oracle Spatial 网络数据模型
Oracle 技术白皮书
作者：Jack Chenghua Wang

Oracle 公司全球总部
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

电话 650.506.7000
传真 650.506.7200
国际咨询：
电话 44.932.872.020
电报 851.927444(ORACLEG)
传真 44.932.874.625
<http://www.oracle.com>